Intel's Multi-Threading Technology

Hyper-Threading Technology

Glenn Hinton
Intel Fellow

Multi-Threading Research

John Shen
Director, Micro-architecture Lab

Hyper-Threading Technology

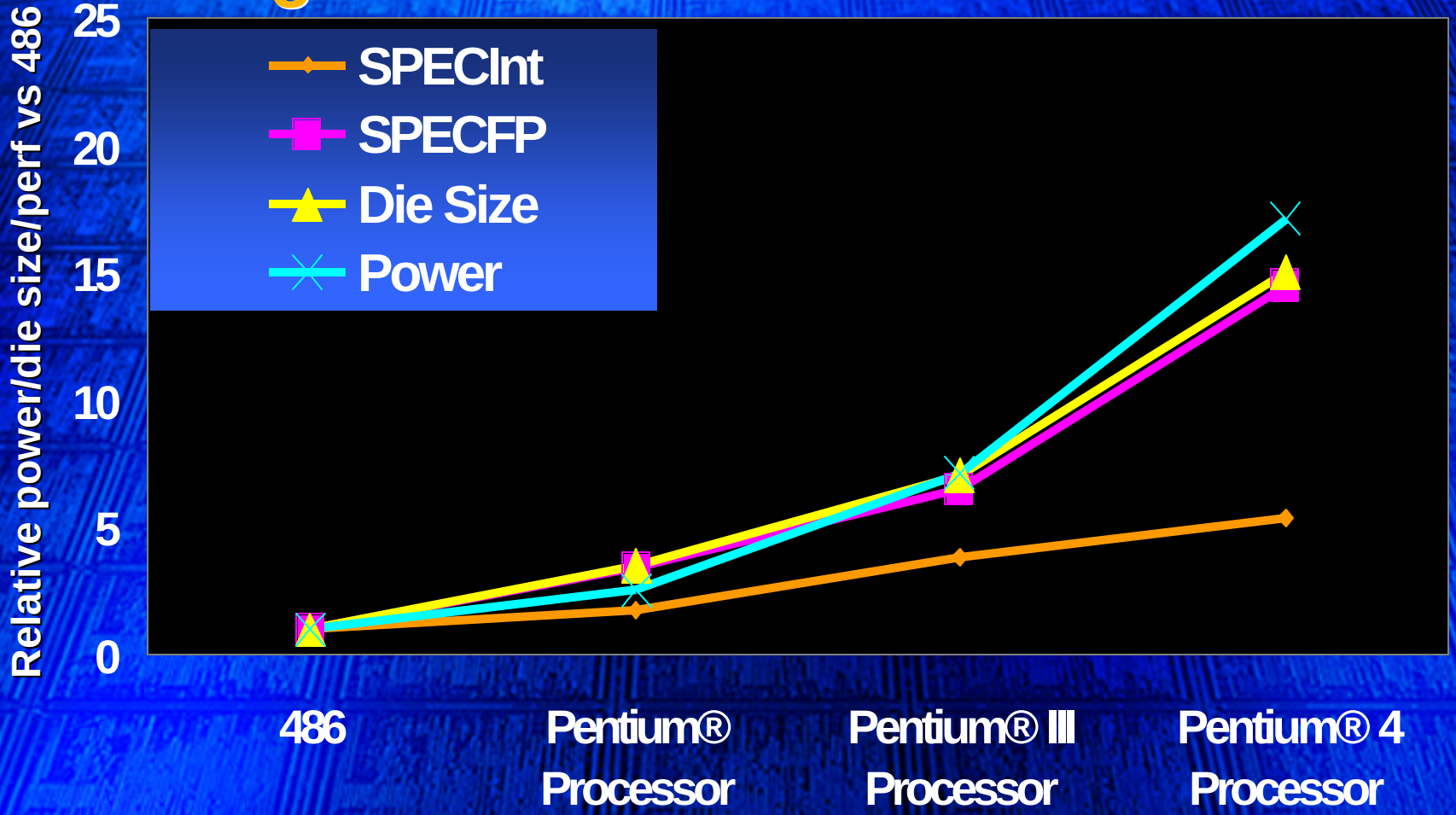
Glenn Hinton
Intel Fellow
Intel Architecture Group
Intel Corporation

October 15, 2001

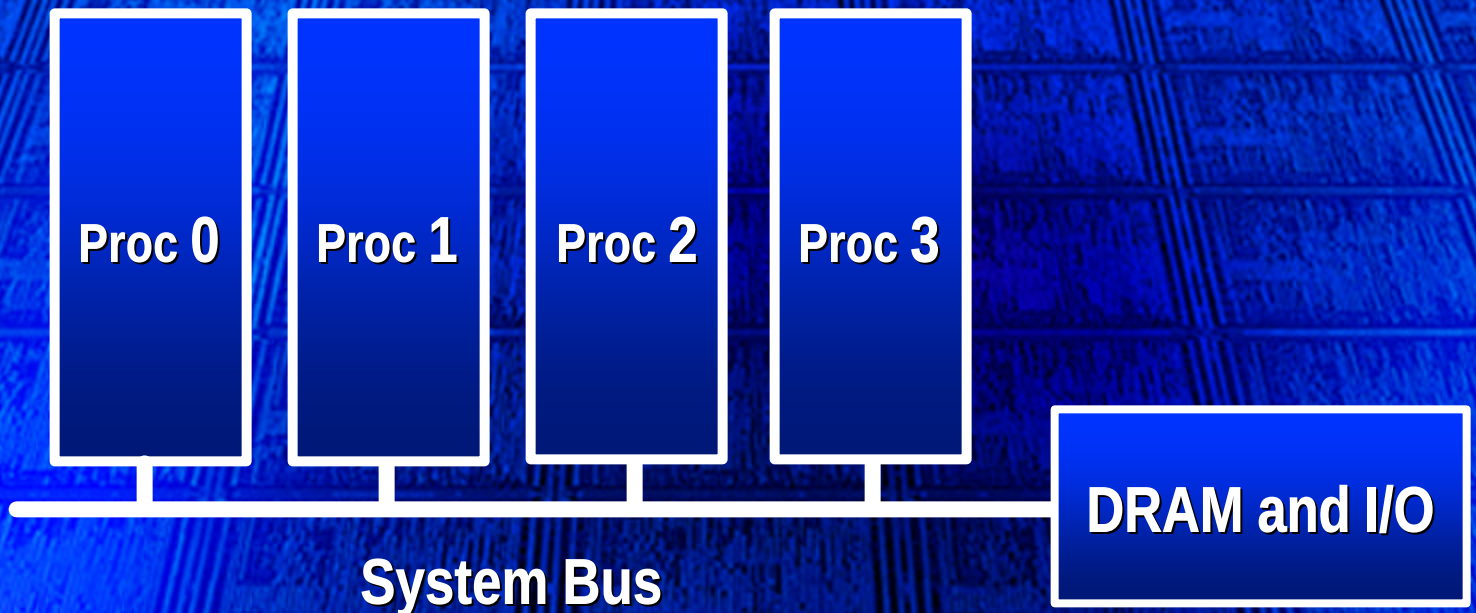
Outline

- Motivation for Multi-threading
- "Thread-Level Parallelism" Optimized HW
- Hyper-Threading Implementation
- Initial Performance Results

Single-stream Performance vs Costs



Multi-Processor Systems: Thread Level Parallelism



Parallelism Design Space

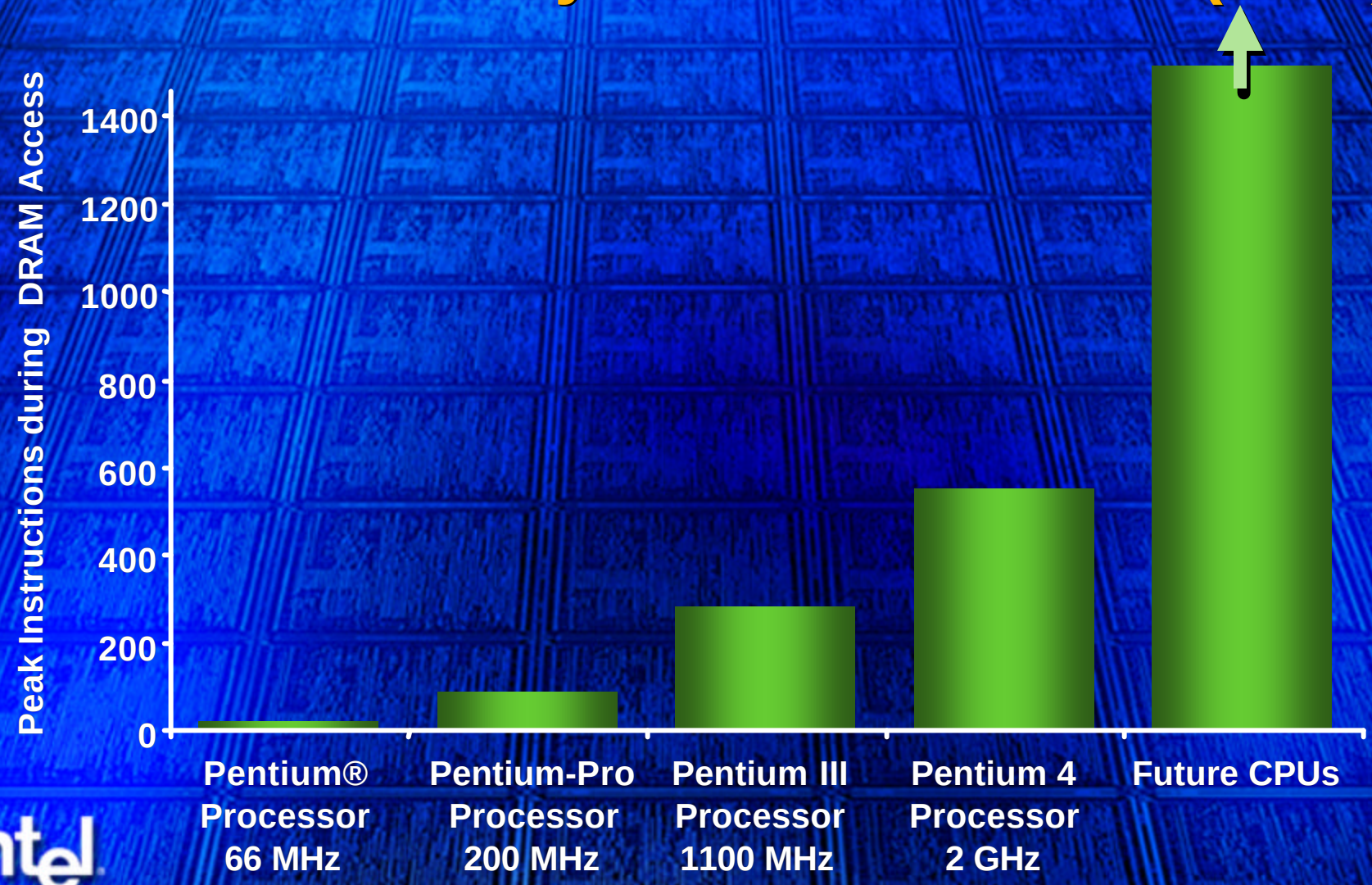
Single-Stream Perf

Faster frequency
Wider Superscalar
More Out-of-order speculation

Multi-Stream Perf

More CPUs
Multi-Core

Long Latency DRAM Accesses: Needs Memory Level Parallelism (MLP)



Today's Software

- Servers
  - Multi-threaded and highly scalable on multi-processor systems
- High-end desktop and workstation
  - Increasingly multi-threaded
- Desktop
  - Increasing number of support threads
  - Simultaneously running multiple unrelated applications
  - Windows XP* allows MP in Desktop

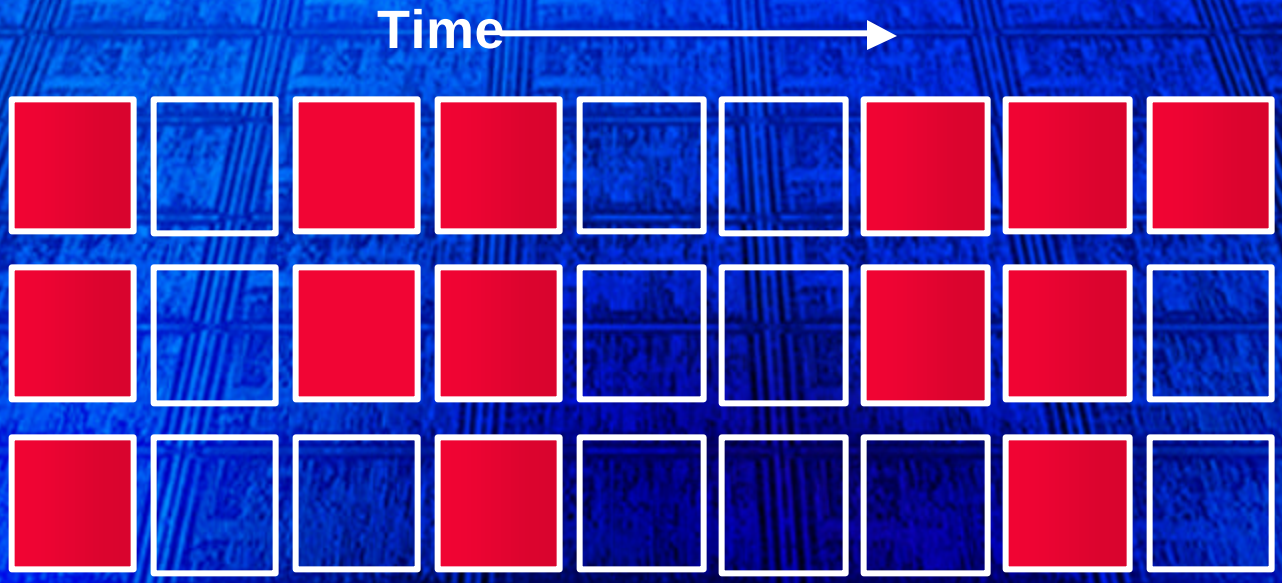
Opportunity to Exploit Thread-level Parallelism In Today's Software and Usage Models

*Third-party brands and names are the property of their respective owners.

Outline

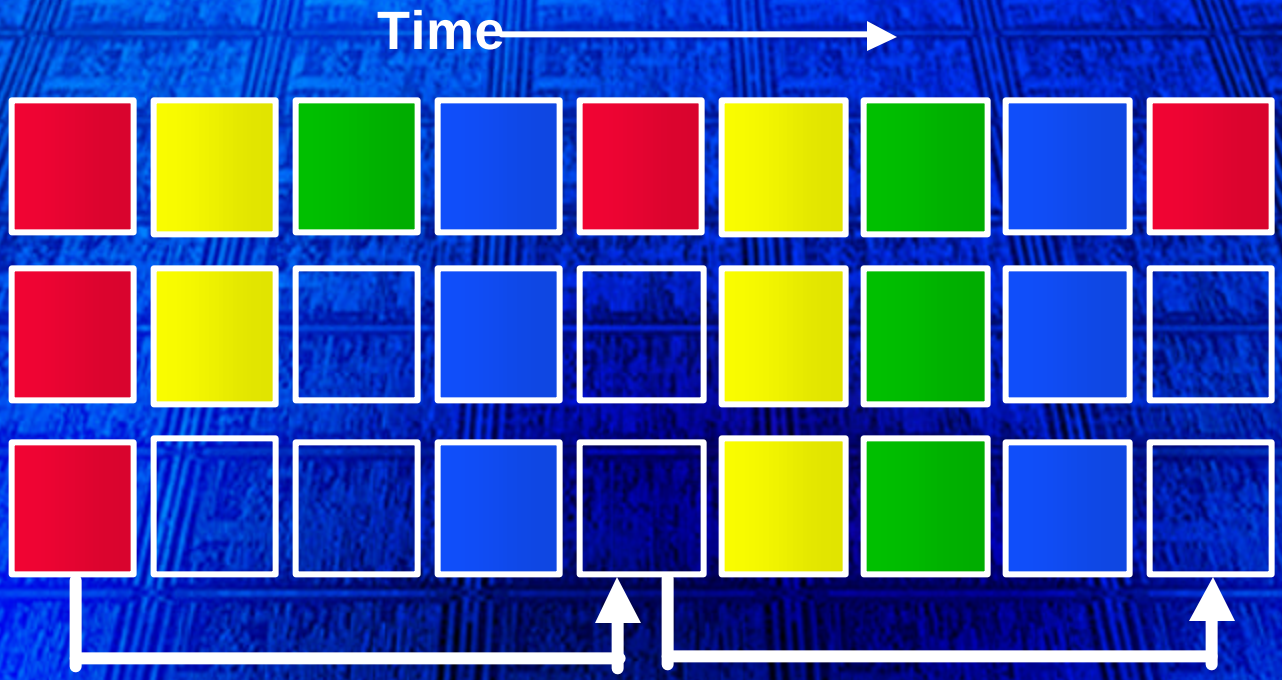
- Motivation for Multi-threading
- **'Thread-Level Parallelism' Optimized HW**
- Hyper-Threading Implementation
- Initial Performance Results

Superscalar Issue

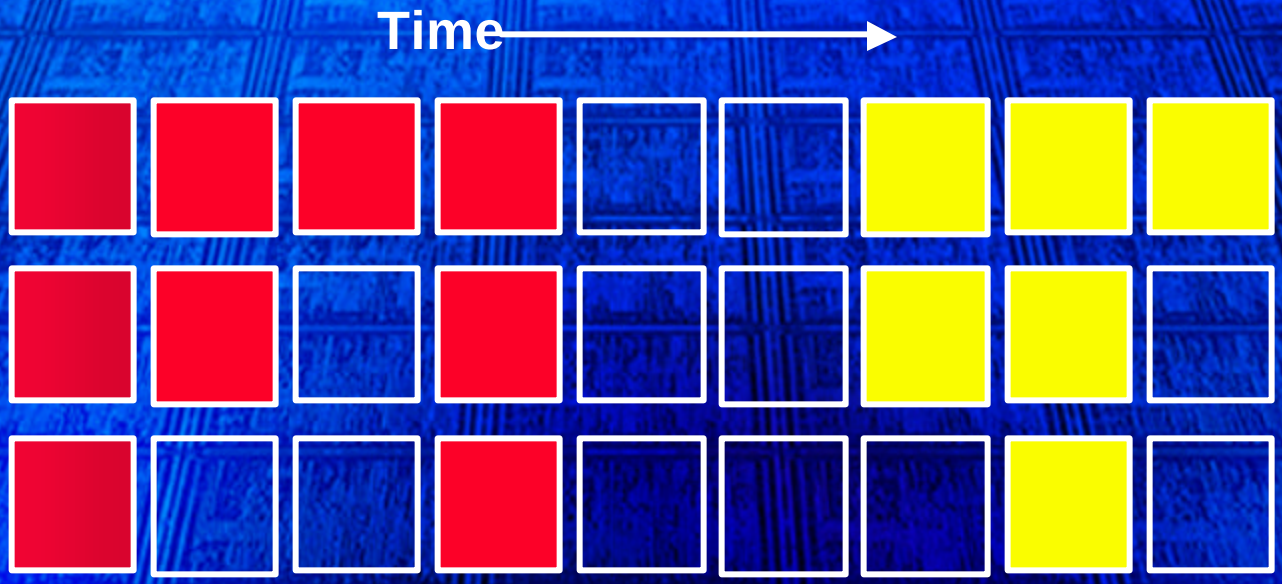


Chip Multiprocessor (CMP)

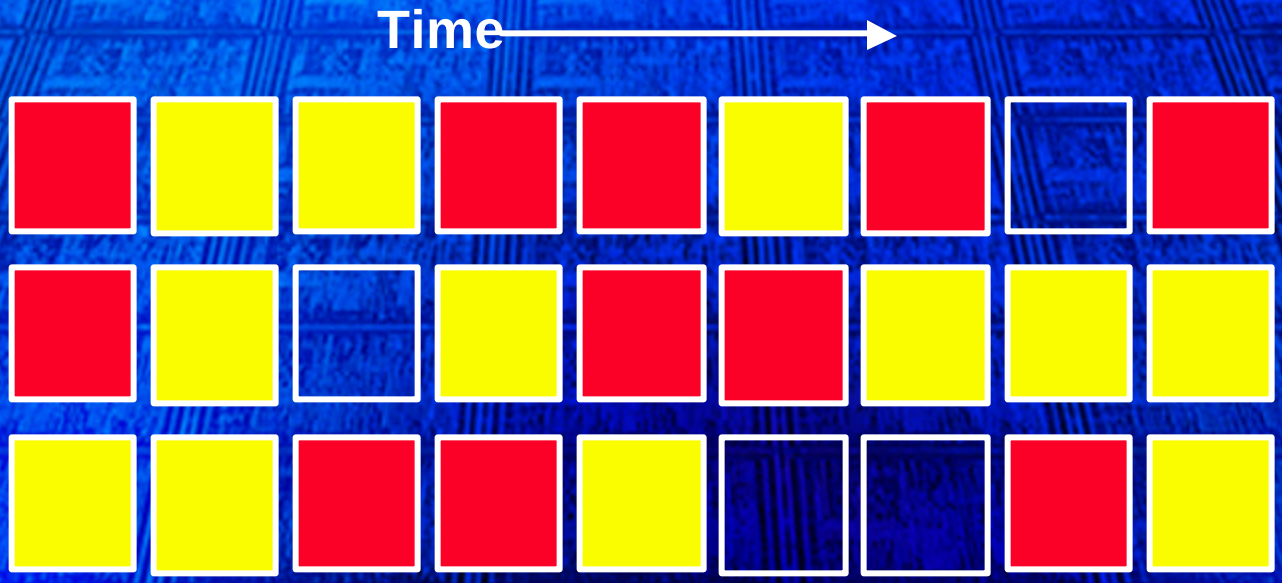
Fine Grained Time-slicing Multi-Threading



Switch-on-Event Multi-Threading (SOE-MT)



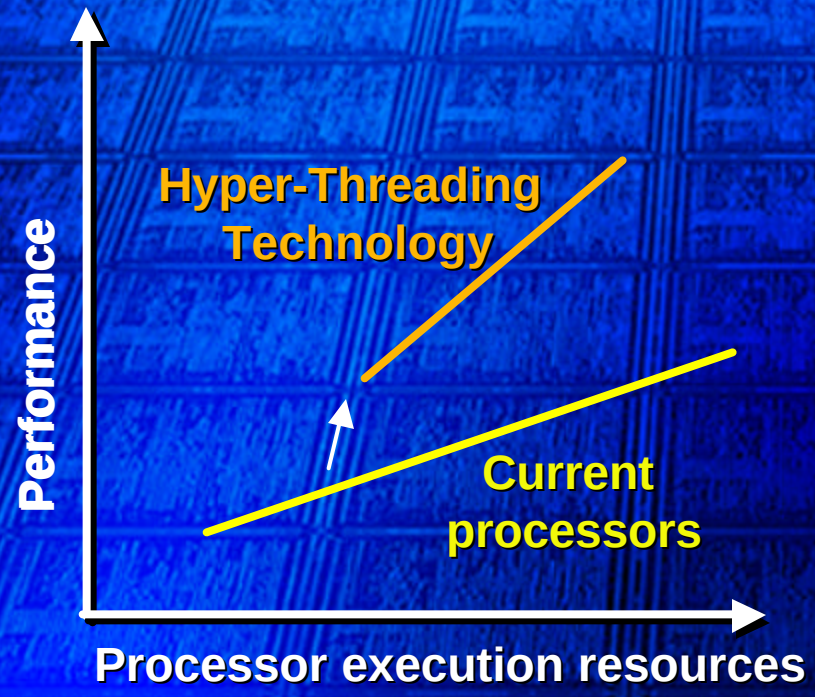
Simultaneous Multi-Threading (SMT)



Maximum utilization of function units by independent operations

intel

Multi-Threading Accelerates Performance of Threaded Applications



- SMT: most efficient/highest performance option
- More performance for CPU execution resources
  - Uses resources more fully
- Greater performance improvements from additional processor resources

Outline

- Motivation for Multi-threading
- 'Thread-Level Parallelism' Optimized HW
- **Hyper-Threading Implementation**
- Initial Performance Results

Hyper-Threading Technology is SMT

- Executes two tasks simultaneously
  - Two different applications
  - Two threads of same application
- CPU maintains architecture state for two processors
  - Two logical processors per physical processor
- Demonstrated on prototype Intel® Xeon™ Processor MP
  - Two logical processors for < 5% additional die area
  - Power efficient performance gain
  - Result of significant research, design effort, and validation

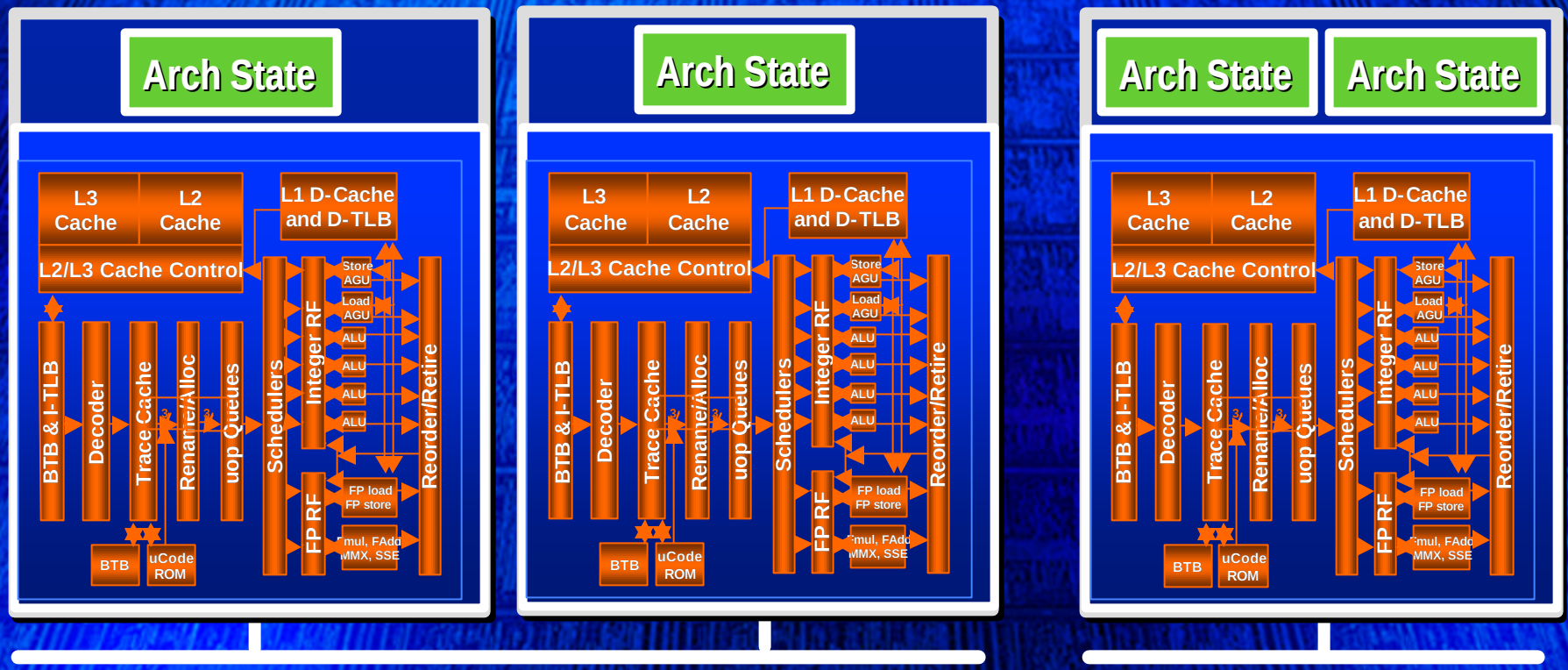
Hyper-Threading Technology brings Simultaneous Multi-Threading(SMT) to Intel Architecture

intel

Resources: Replicated vs Shared

Multiprocessor

Hyper-Threading

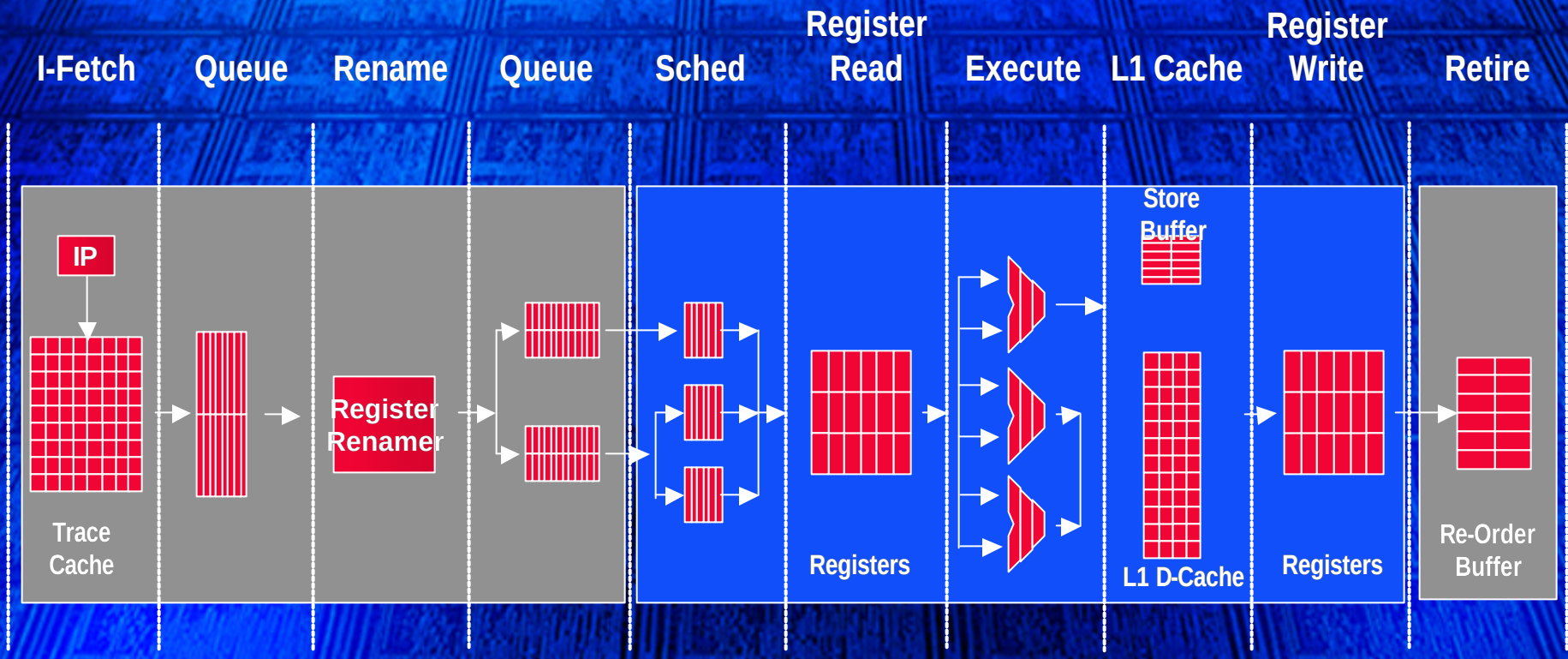


Multi-Processors replicate execution resources
Hyper-Threading Technology shares resources

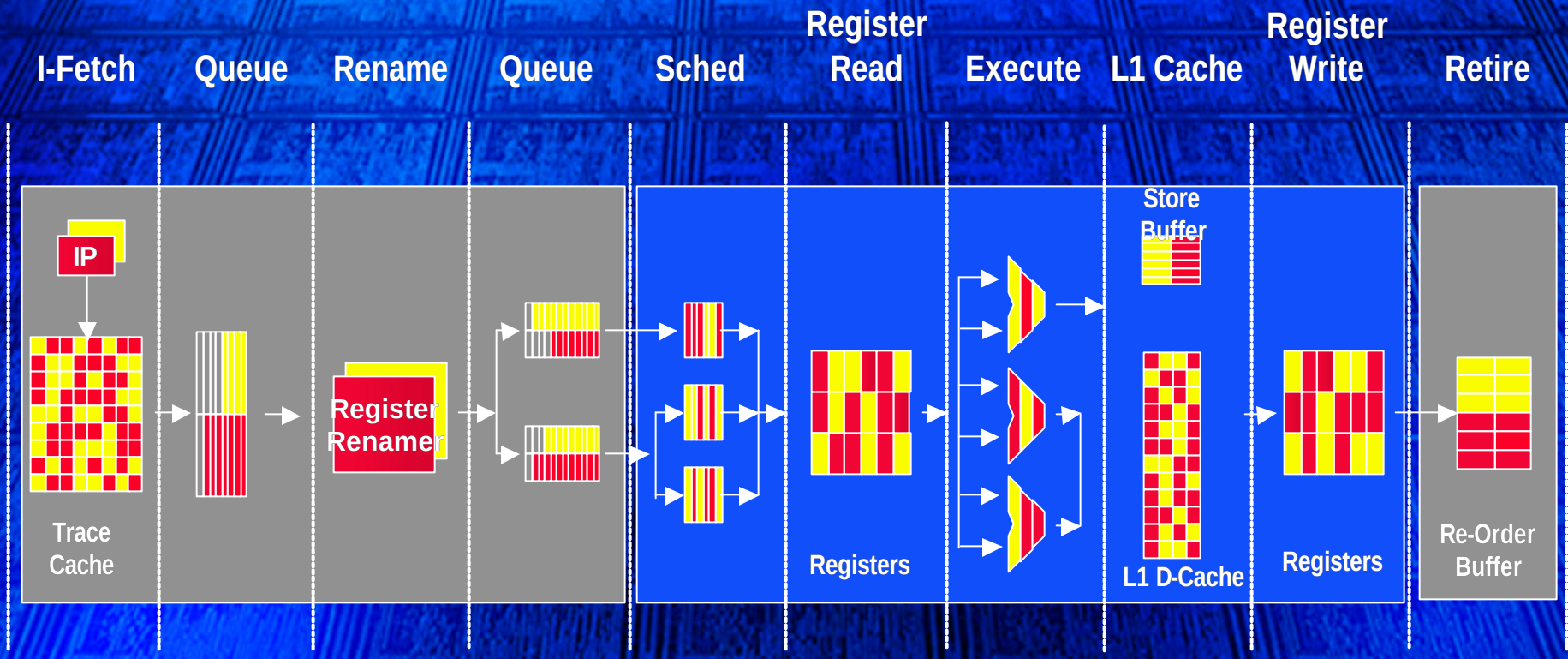
Changes for Hyper-Threading

- Replicate resources
  - All per-CPU architectural state
  - Instruction Pointers, renaming logic
  - Some smaller resources
    - E.g, return stack predictor, ITLB, etc
- Partition resources (share by splitting in half per thread)
  - Several buffers (Re-order buffer, load/store suffers, queues,etc)
- Share most resources
  - Out-of-Order execution engine
  - Caches

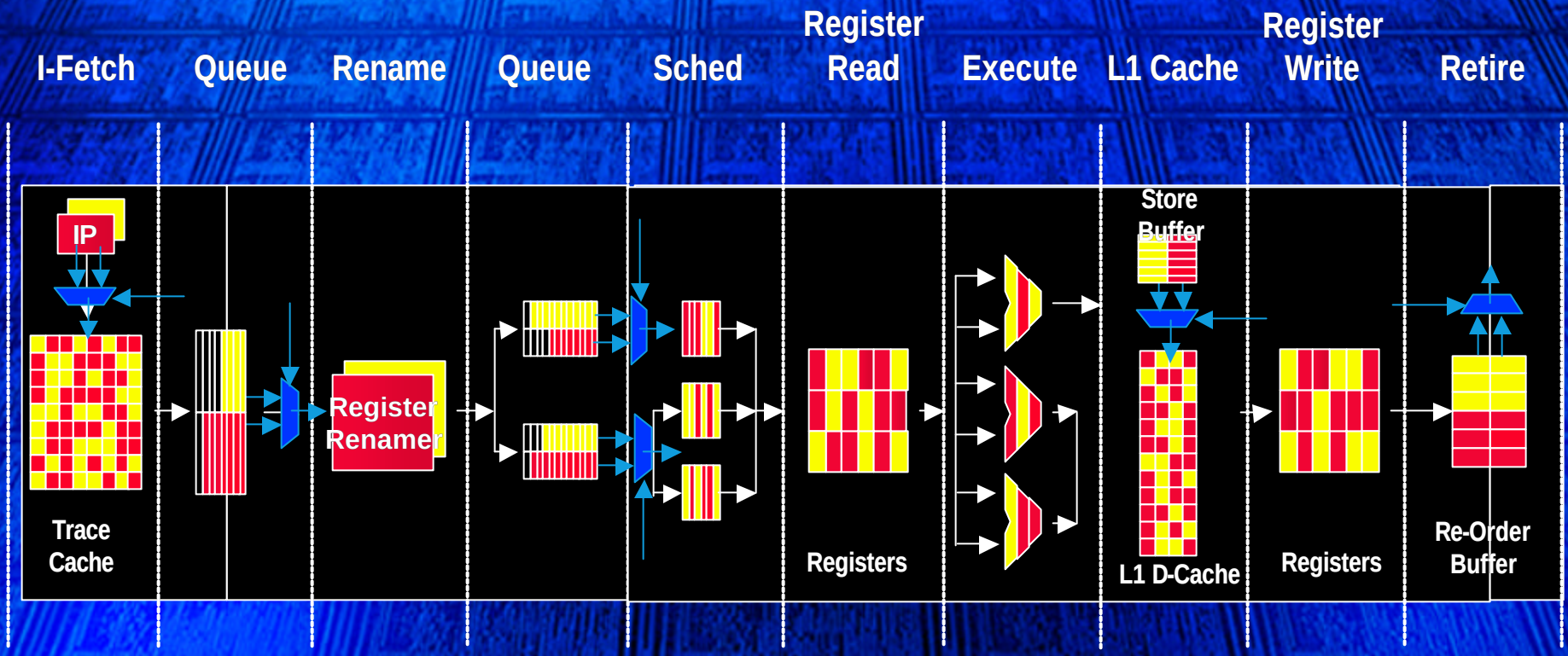
Out-of-Order Execution Pipeline



Hyper-Threading Pipeline

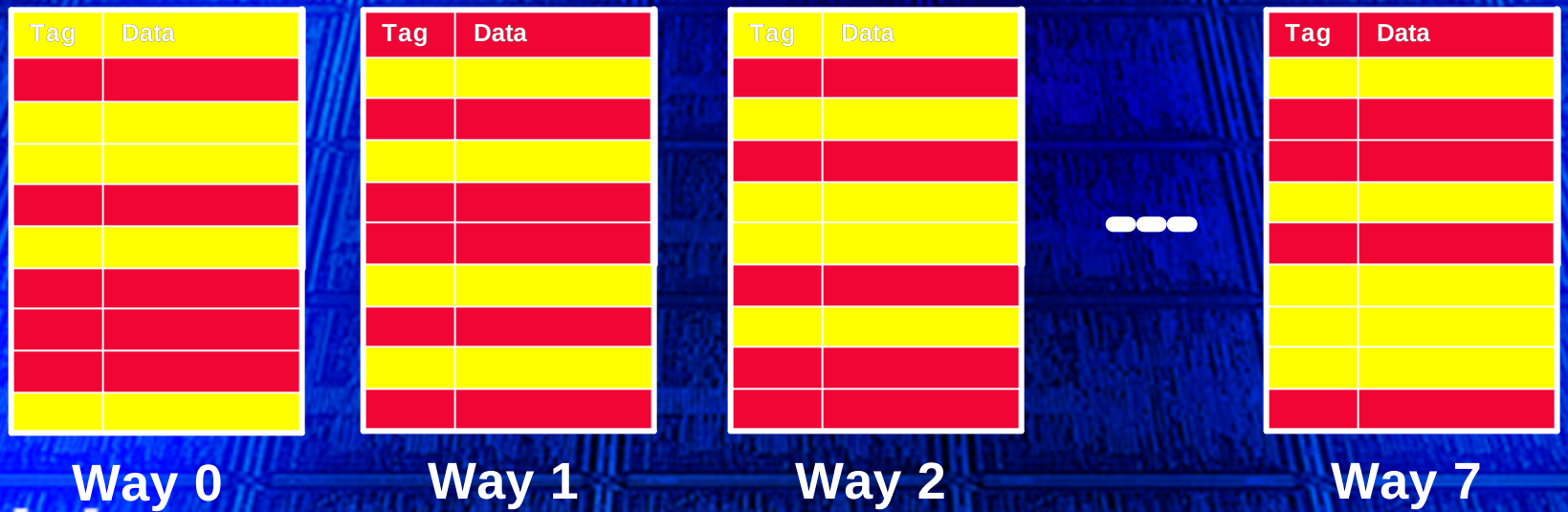


Thread-Selection Points



Caches Shared

- Execution Trace Cache
- First-level Data Cache
- Second-level Unified Cache
- Third-level Unified Cache

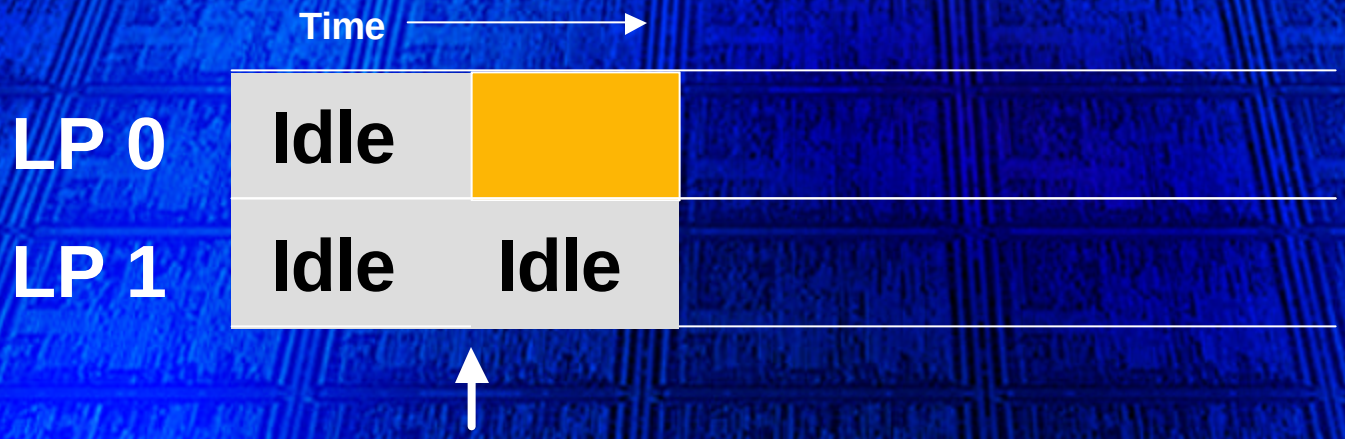


Data in Caches can be Shared

- First-level Data Cache
- Second-level Unified Cache
- Third-level Unified Cache

Operating System

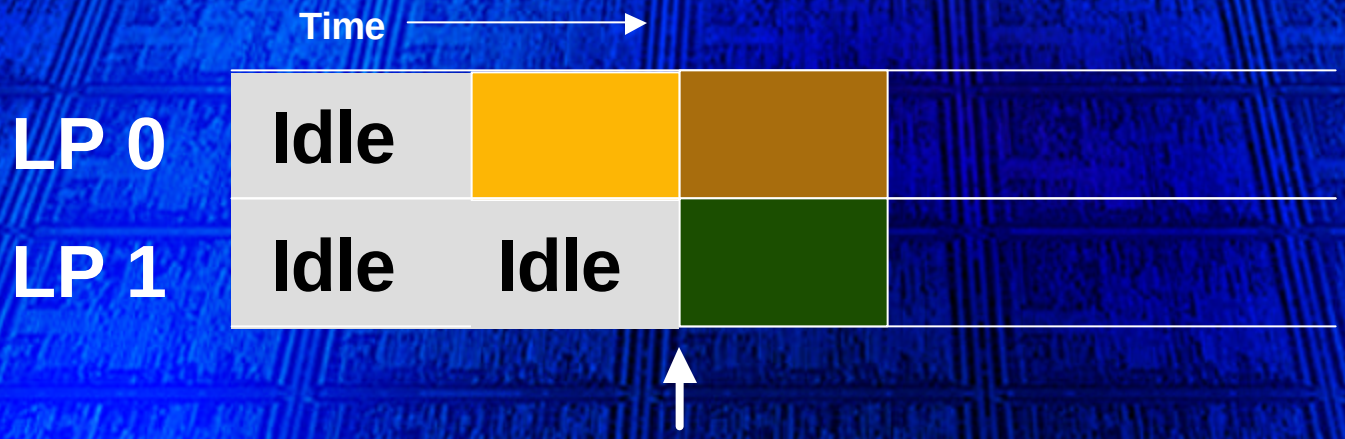
- Operating system manages tasks
  - Schedule task on logical processors
  - Execute HALT if one logical processor idle



O/S schedules a task on LP 0

Operating System

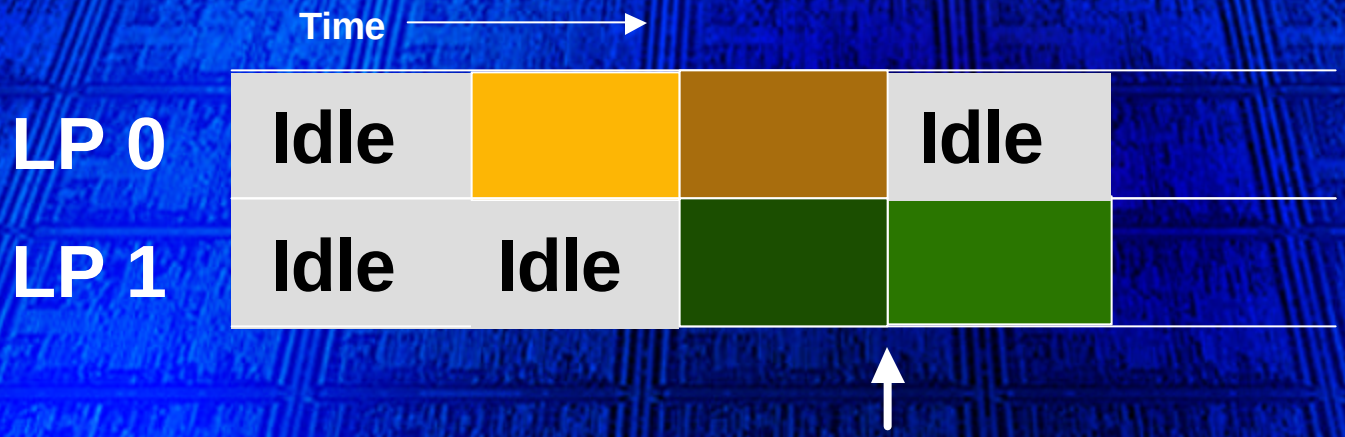
- Operating system manages tasks
  - Schedule task on logical processors
  - Execute HALT if one logical processor idle



O/S schedules a task on LP 1
Both logical processors active

Operating System

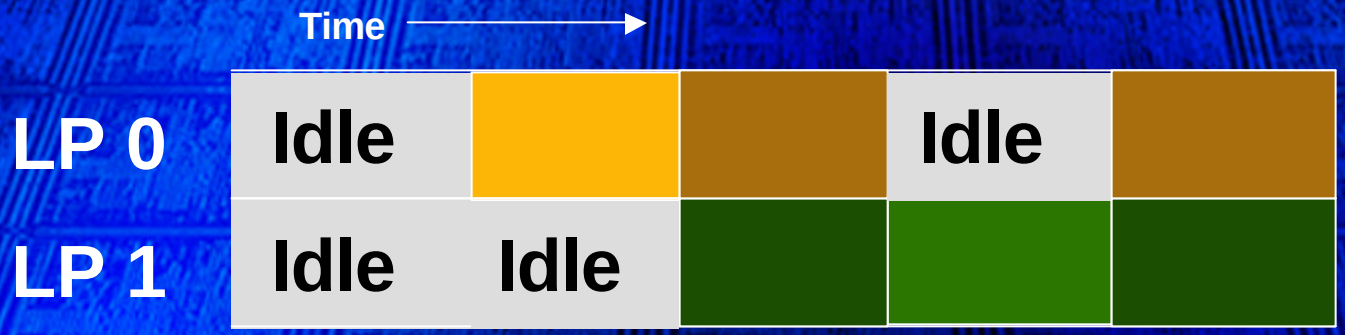
- Operating system manages tasks
  - Schedule task on logical processors
  - Execute HALT if one logical processor idle



Task on LP 0 finishes,
O/S executes HALT on LP0

Operating System

- Operating system manages tasks
  - Schedule task on logical processors
  - Execute HALT if one logical processor idle



Outline

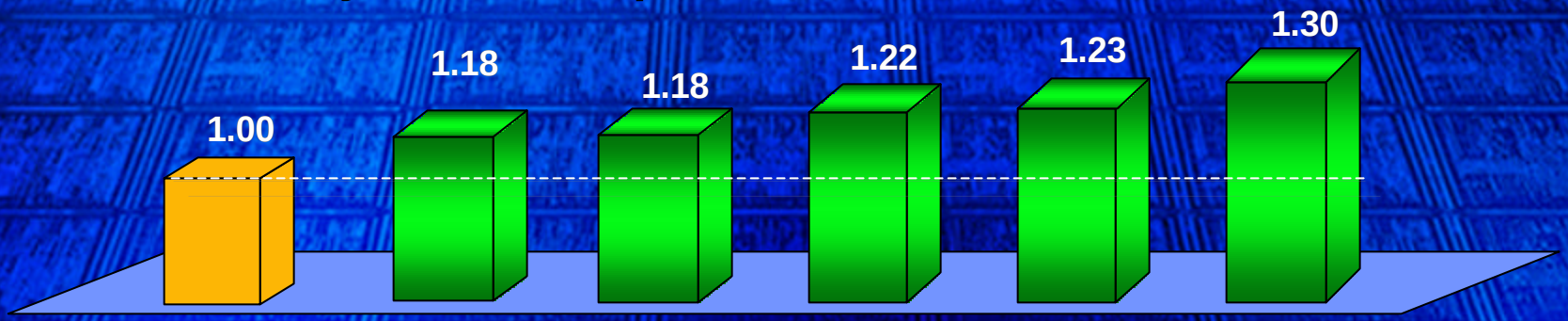
- Motivation for Multi-threading
- 'Thread-Level Parallelism' Optimized HW
- Hyper-Threading Implementation
- **Initial Performance Results**

Performance Results

- Preliminary performance on a few applications
  - All applications unmodified
- Continuing evaluation over larger pool of applications
- Performance varies as expected with:
  - Number of parallel threads
  - Resource utilization

Intel® Xeon™ Processor MP
Hyper-Threading Technology Performance

Application Measurements
by Intel Microprocessor Software Labs



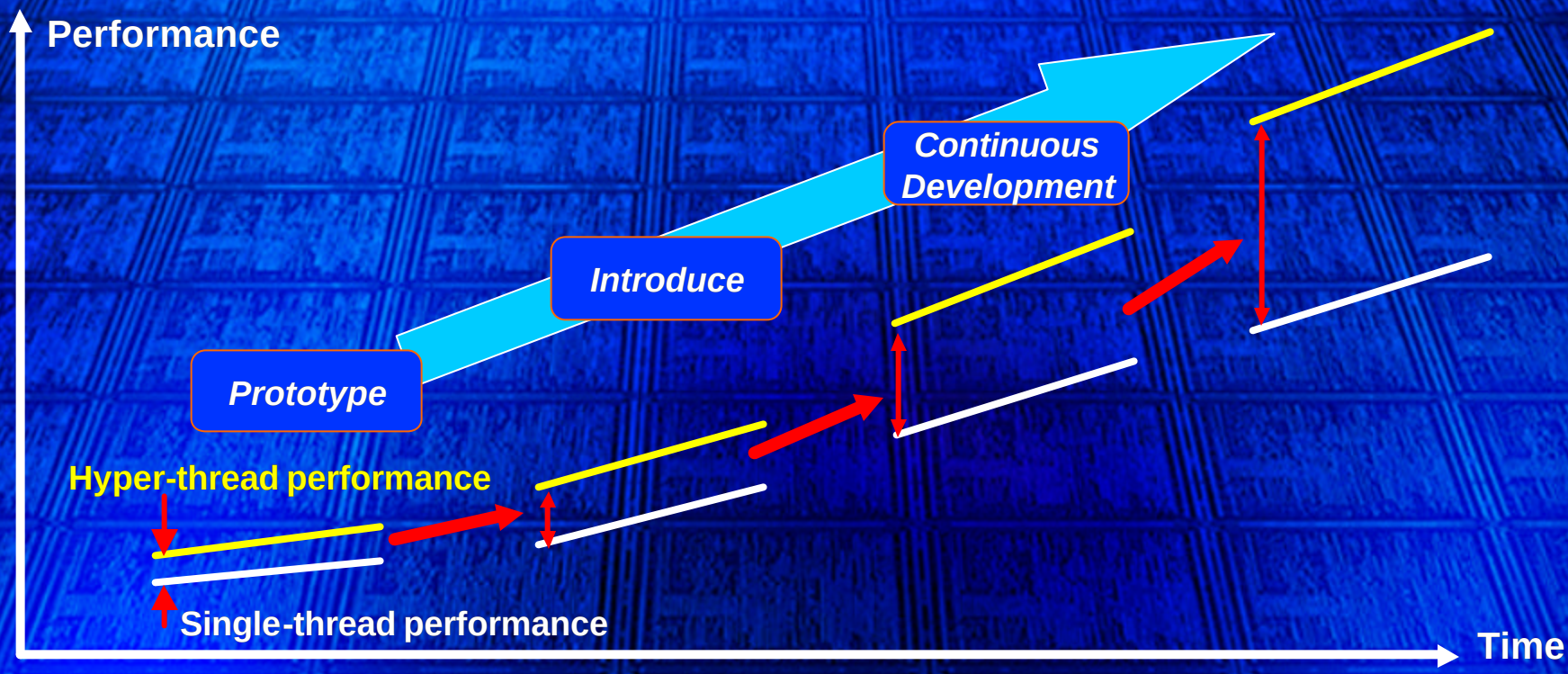
Preliminary Hyper-Threading Technology performance numbers on prototype Intel Xeon processor MP platforms today

Source: Intel Microprocessor Software Labs, Intel Xeon Processor MP (1.6GHz, 1M iL3 cache) platforms are prototype systems in 2-way configurations. Applications not tuned or optimized for Hyper-Threading Technology

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference www.intel.com/procs/perf/limits.htm or call (U.S.) 1-800-628-8686 or 1-916-356-3104.

*All trademarks and brands are the property of their respective owners

Intel's Long-Term Hyper-Threading Strategy



Hyper-Threading Technology expected to deliver increasingly higher performance

Hyper-Threading Technology Summary

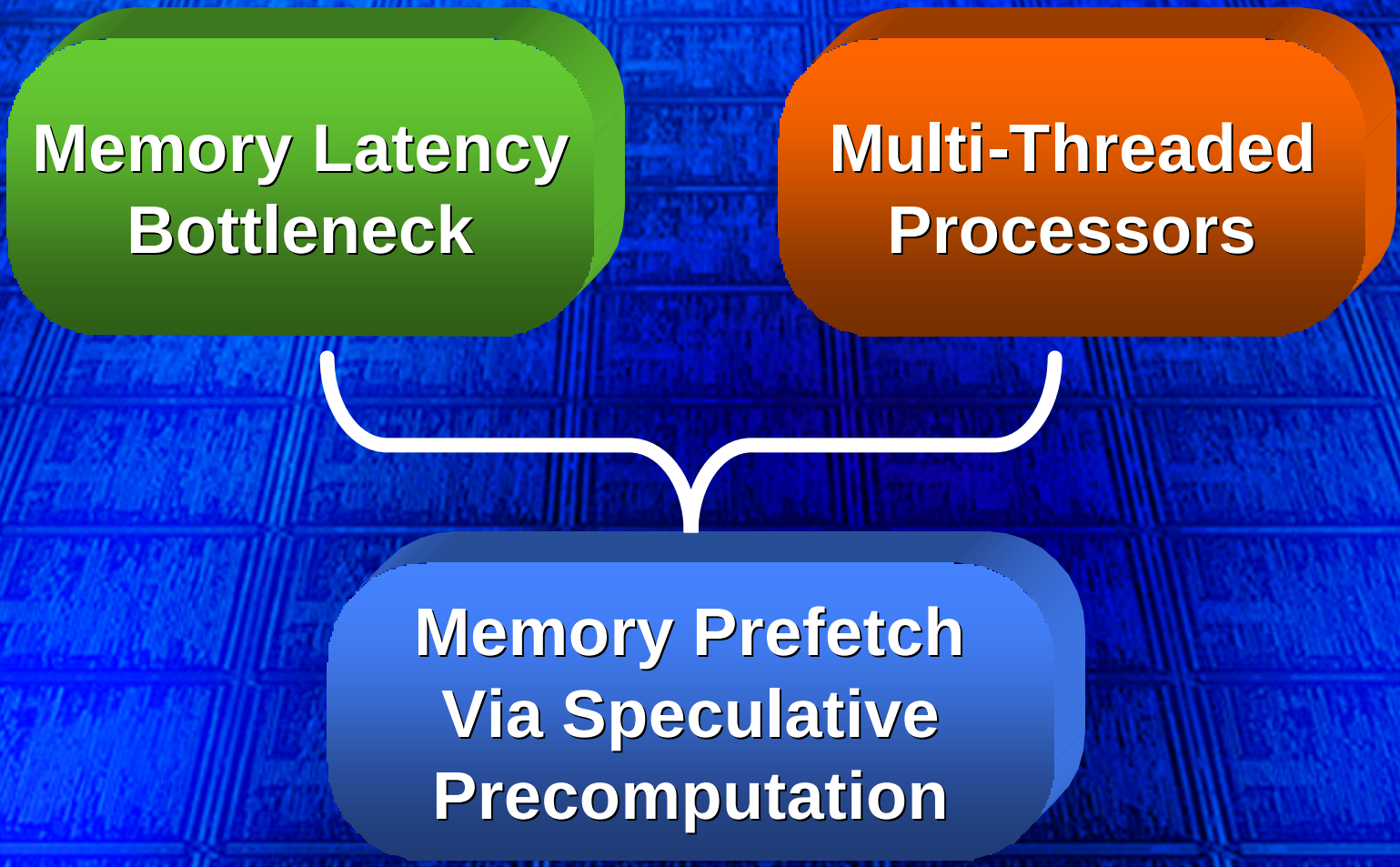
- Prototyped in Intel® Xeon™ Processor MP
- Exploits parallelism in today's applications and usage
  - Two logical processors on one physical processor
- Accelerates performance for low silicon and power costs
- Significant new technology direction for Intel's future CPUs

Multi-Threading Research
"Speculative Precomputation"

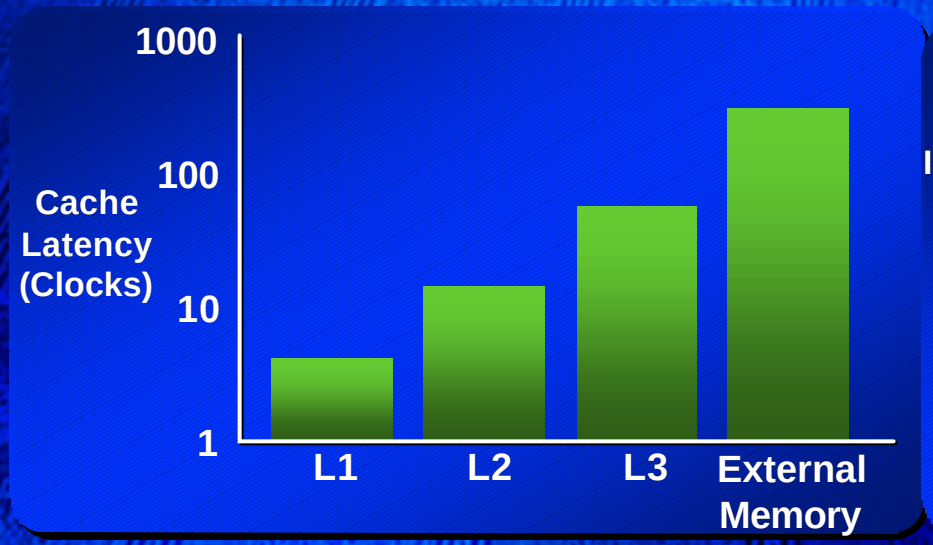
John P. Shen
Director, Microarchitecture Lab
Intel Labs
Intel Corporation

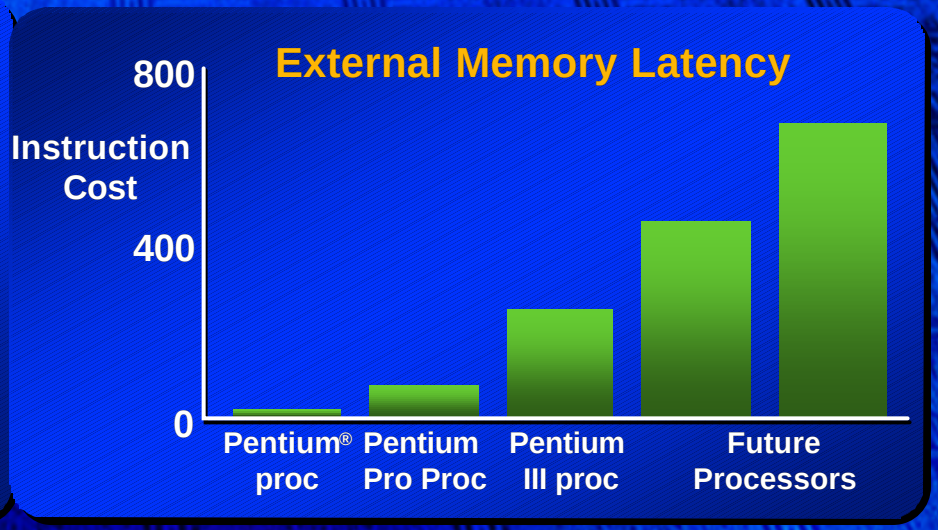
October 15, 2001
Microprocessor Forum

Two Dominant Trends



Memory Latency Bottleneck





Cache Prefetching:

- **Hardware:** Limited by predictable patterns
- **Software:** Limited by single control flow
- **Research Challenge:** Pointer-intensive code

Source: Intel

Multi-Threaded Processors

- **Targeting:**
  - Throughput of Multi-tasking Workloads
  - Latency of Multi-threaded Applications
- **Not Targeting:**
  - Latency of Single-threaded Applications
- **Research Challenge:**
  - Leverage Multi-threaded CPU to Improve Latency of Single-threaded Applications

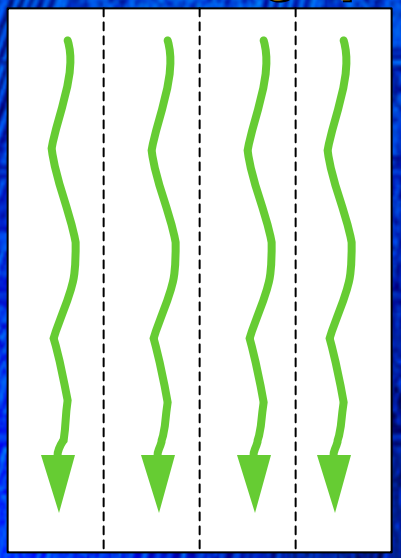
“Pseudo Multi-Threading”

- Speculatively execute special assist threads in available thread contexts
- Assist threads are attached to original binary without requiring source code recompilation
- Leverage execution of assist threads to speed up original single-threaded program

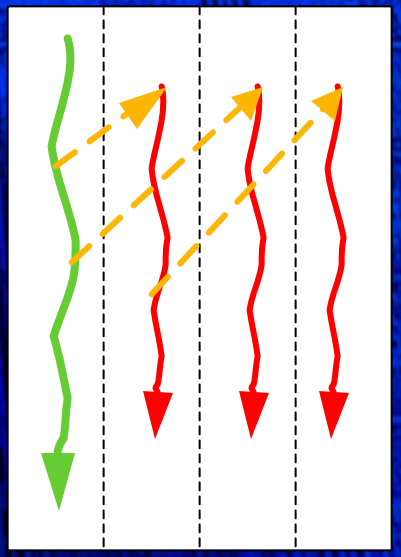
New Opportunities for Parallelism

Pseudo MT Illustrated

Multi-threading for throughput



Multi-threading for latency



Non-speculative main thread

Speculative threads

Thread spawning

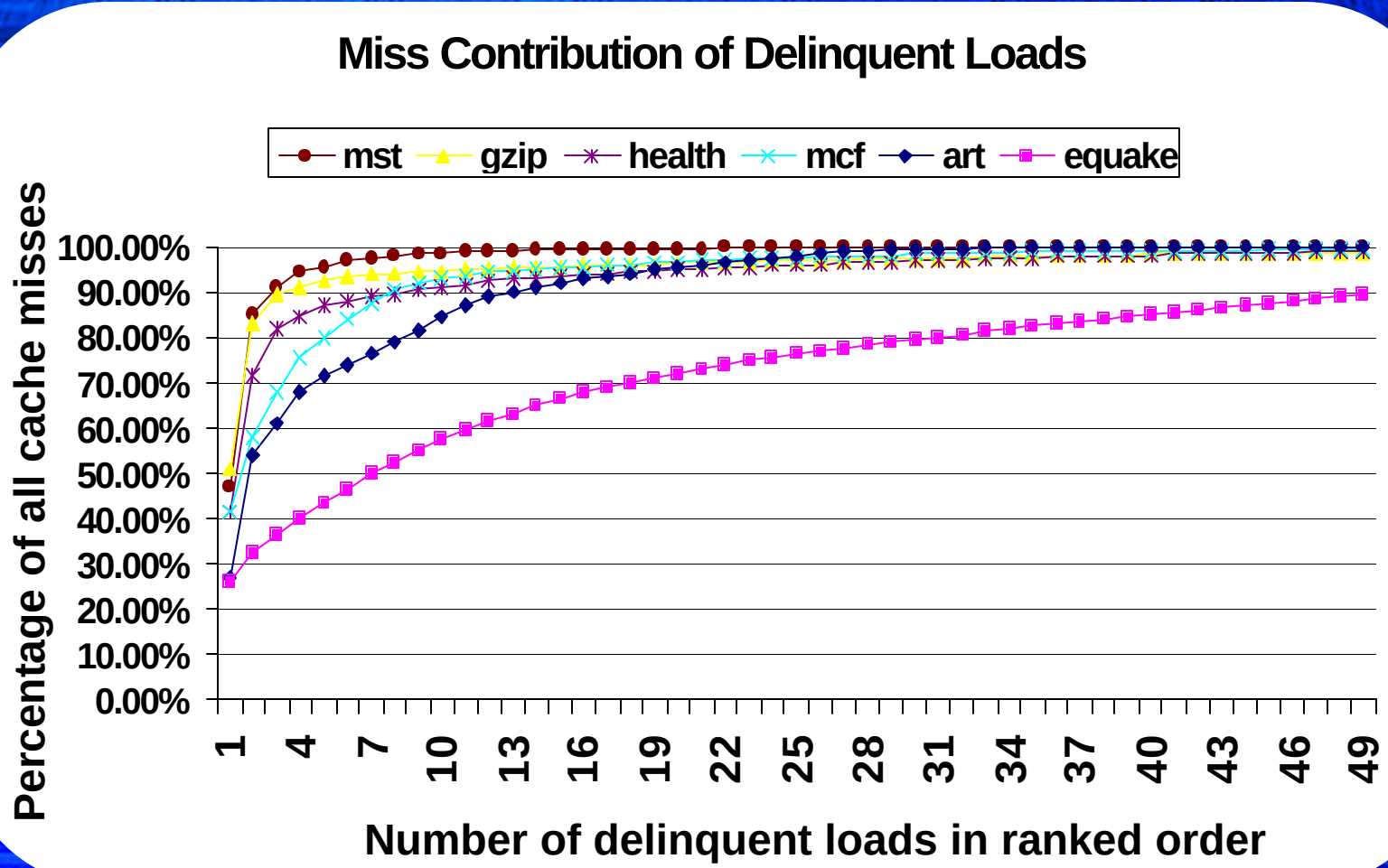
Speculative Precomputation

- **Target:** The Memory Bottleneck
  - Pointer-intensive applications
  - Pre-fetch for “Delinquent loads”
- **Method:** A Form of Pseudo MT
  - Embed pre-fetching SP threads in binary
  - Parallel execution of main and SP threads

Eliminate and reduce stall cycles due to Performance-degrading cache misses

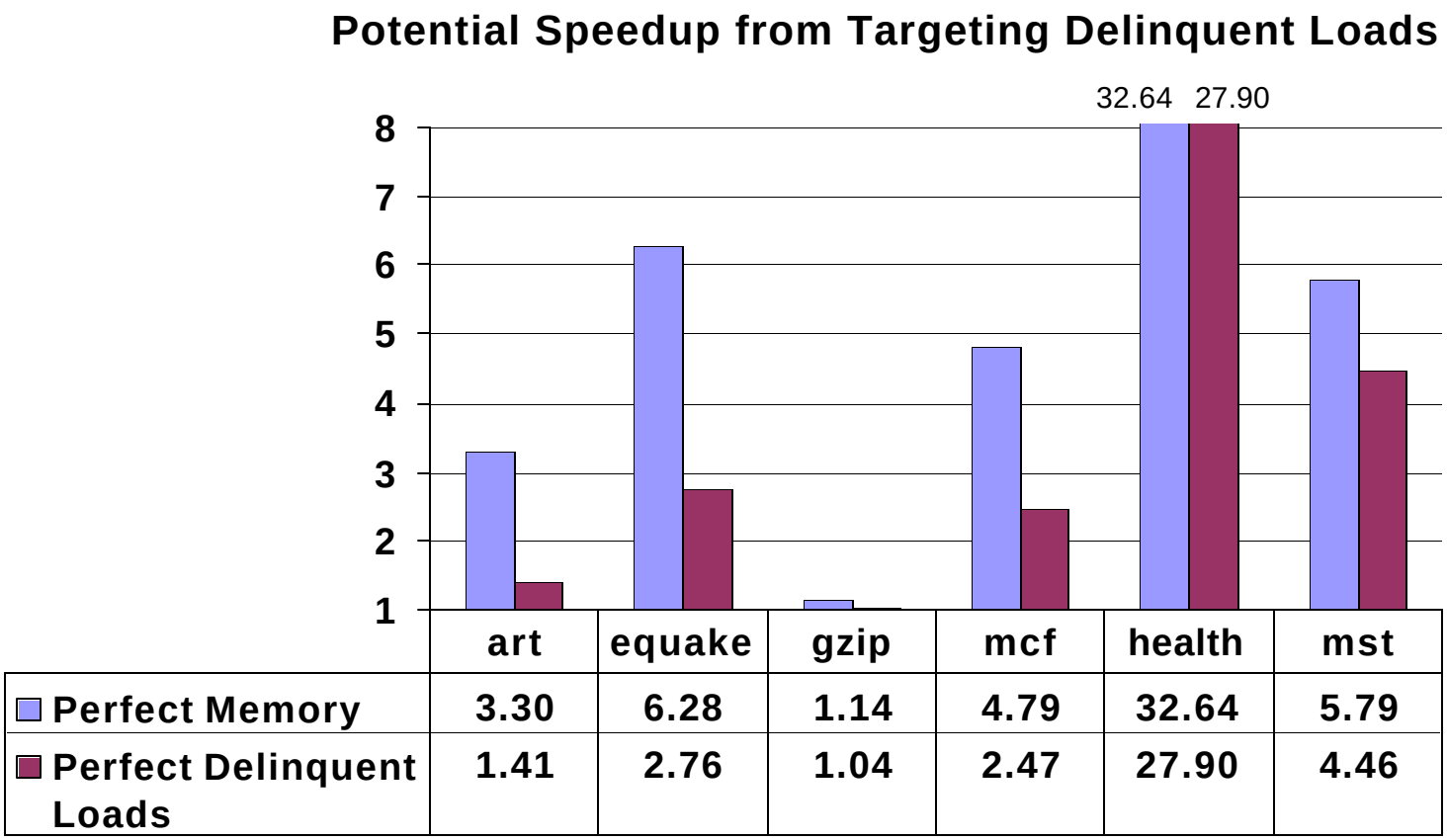
intel

Delinquent Loads



Source: Intel

Importance of D-Loads



	art	equake	gzip	mcf	health	mst
Perfect Memory	3.30	6.28	1.14	4.79	32.64	5.79
Perfect Delinquent Loads	1.41	2.76	1.04	2.47	27.90	4.46

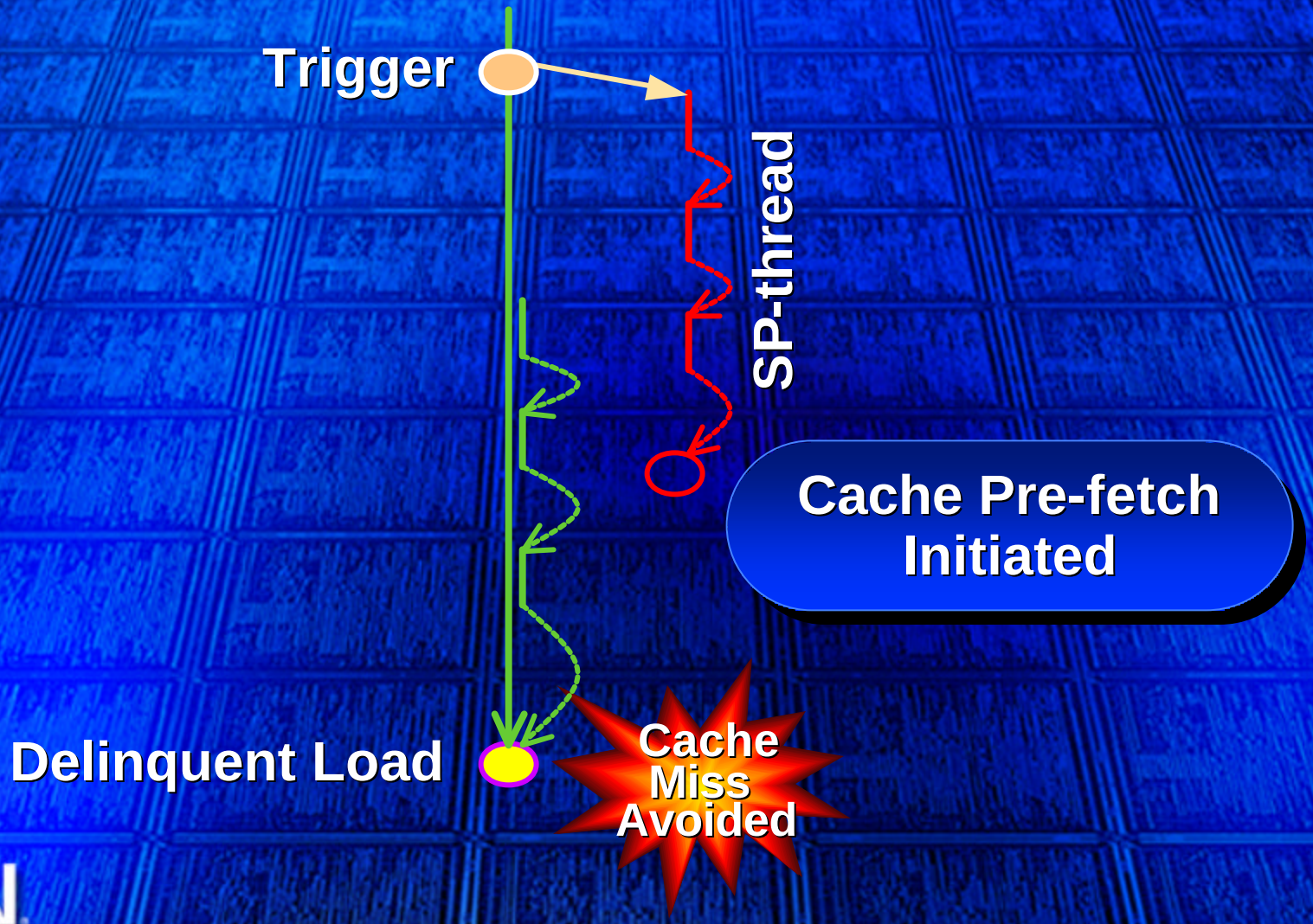
Source: Intel

SP-Threads Statistics

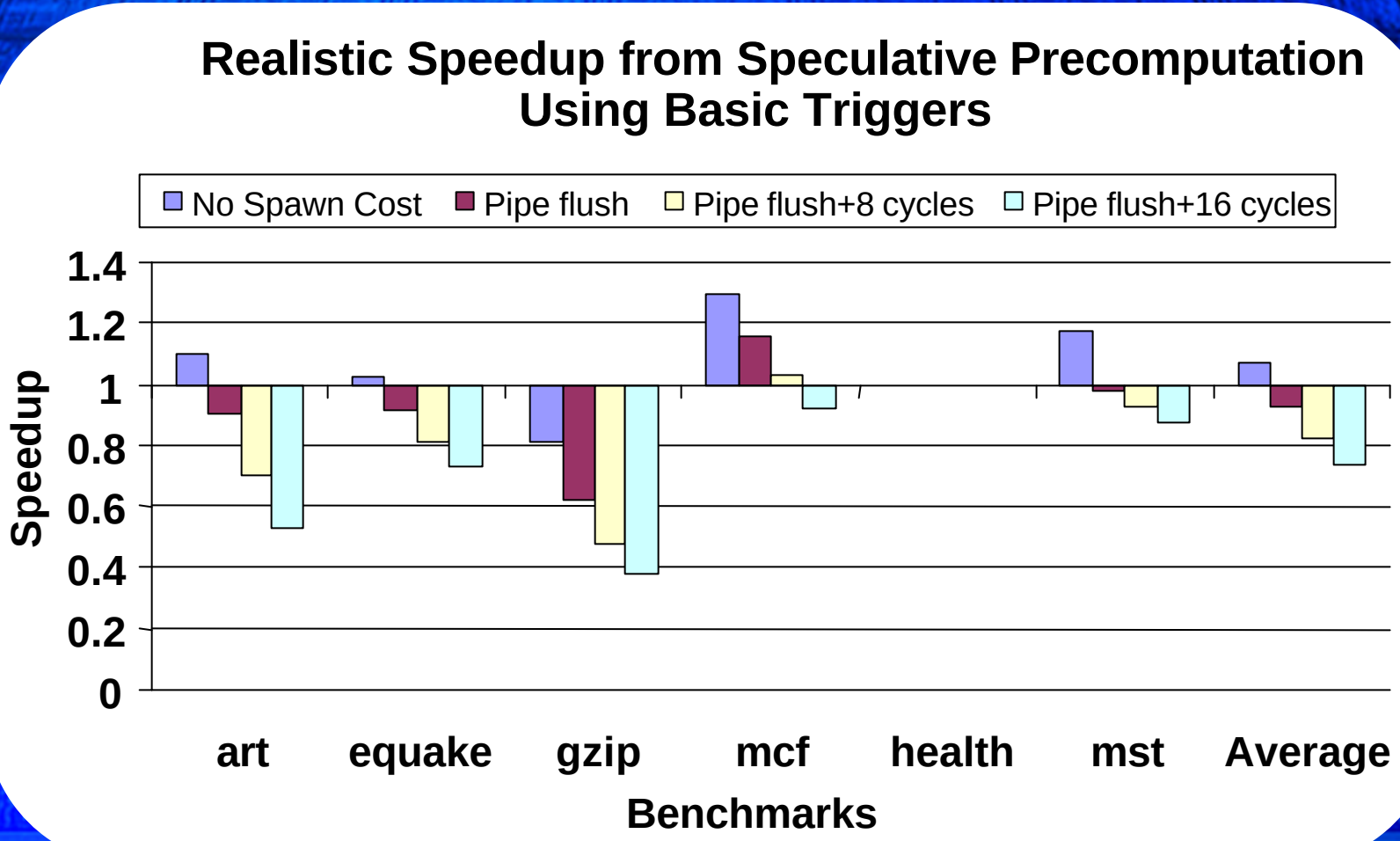
Statistics on SP-Threads for top 10 Delinquent Loads			
Benchmark	# of SP Threads	Avg. Thread Length	Avg. # of Live-Ins
art	2	4	3.5
equake	8	12.5	4.5
gzip	9	9.5	6.0
mcf	6	5.8	2.5
health	8	9.1	5.3
mst	8	26	4.7

Source: Intel

Pre-fetch via SP-Threads

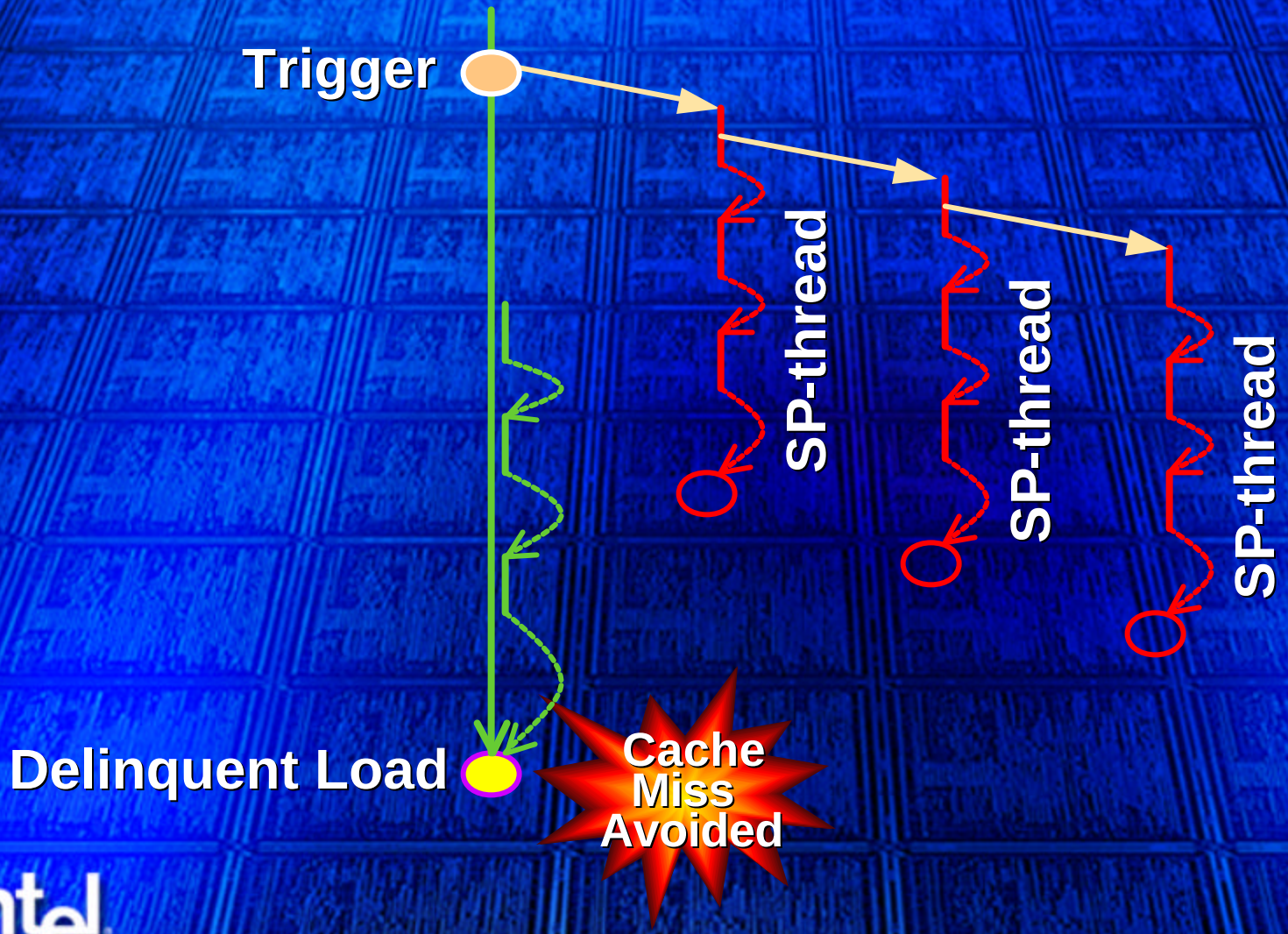


SP Speedup (Basic Triggers)

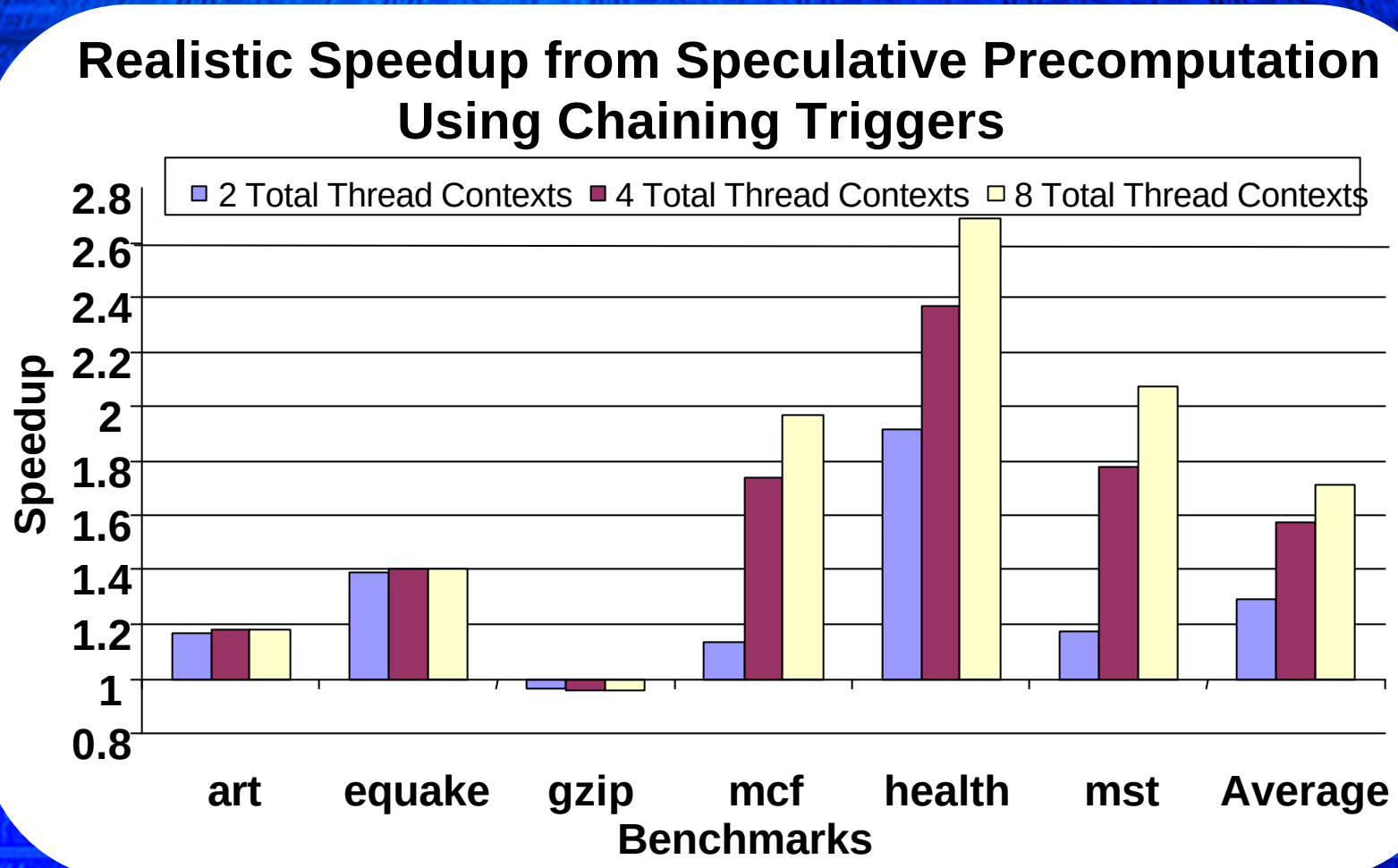


Source: Intel

SP with Chaining Triggers



SP Speedup (Chaining Triggers)



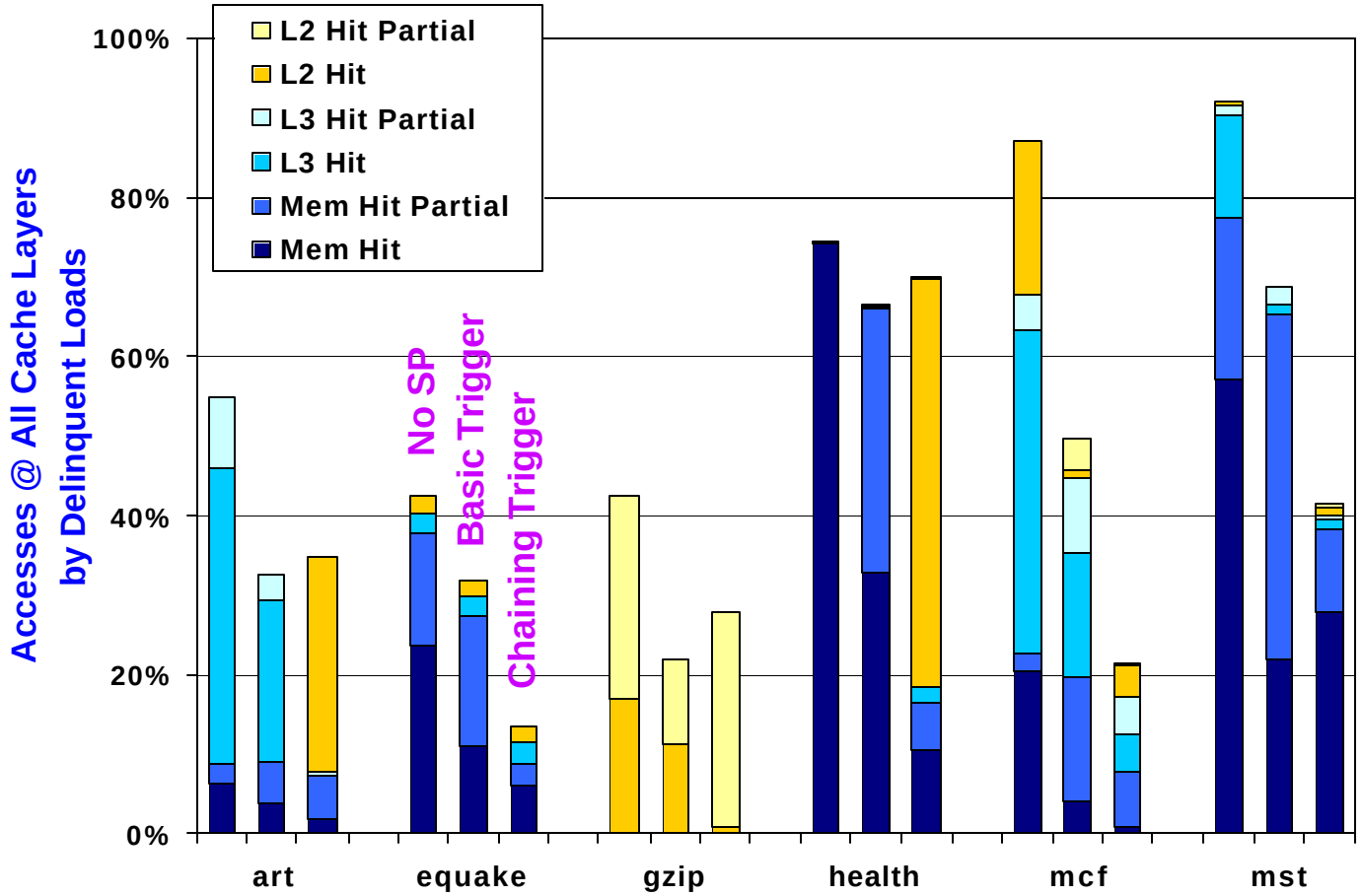
Source: Intel

Chaining Trigger Advantages

- **Low-cost thread spawning:**
  - Chaining triggers initiate SP-threads without impacting main thread performance
- **Long-range prefetching:**
  - Can target delinquent loads far ahead of the main thread
  - Speculative threads make progress independent of main thread's lack of progress
- **Basic triggers initiate precomputation; Chaining triggers sustain precomputation**

intel

Sources of SP Speedup



Source: Intel

Initial HW Experimental Data

Benchmark	Description	Speedup
Synthetic	Graph traversal in large random graph simulating large database retrieval	22% - 45%
MST (Olden)	Minimal Spanning Tree algorithm used for Data Clustering	23% - 40%
Health (Olden)	Hierarchical database modeling health care system	11% - 24%
MCF (SPECint)	Integer programming algorithm used for bus scheduling	7.08%

Source: Intel

Future Research Directions

- **Code Adaptation Tools**
  - Automated Source Level Tools
  - Automated Binary Level Tools
- **Run-Time Support Optimization**
  - Light-Weight Thread Spawning
  - ISA Extension and Microcode Support
- **Dynamic Speculative Precomputation**
  - Microarchitecture Support [w/ UCSD]

intel